REMARKS

This amendment is filed in response to the Office Action dated March 7, 2007. By these amendments, claims 13, 25, and 29 have been amended to address non-statutory subject matter. Claims 2-5 and 14-17 have been amended to provide sufficient antecedent basis. Applicants respectfully request reconsideration of all claims and favorable action in this case.

Response to Rejections of Claims at Issue

Claim Rejections under 35 U.S.C.§101:

Claims 13, 25, and 29 were rejected under 35 U.S.C § 101 as being directed to non-statutory subject matter. The Applicants have amended claims 13, 25, and 29 to claim "a computer-readable storage medium tangibly embodying a program of instructions executable by a computer for performing steps to...." The amended claims refer to functional descriptive material that is recorded on a computer-readable medium, and according to MPEP 2106.01, first and second paragraphs, the amended claims are considered statutory. Accordingly, the Applicants believe that the 35 U.S.C.§101 rejections have been addressed.

Claim Rejections under 35 U.S.C. §112, second paragraph:

Claims 3-4 and 14-15 were rejected under 35 U.S. C. §112, second paragraph, as having insufficient antecedent basis. The Applicants have amended claims 2-5 and 14-17 to provide sufficient antecedent basis.

Claim Rejections under 35 U.S.C. §102(b):

General remarks:

Claims 1-31 were rejected as being anticipated by Microsoft ("ASP.NET QuickStarts Tutorial", dated 2002, hereinafter referred to as "App_trace"). The pages from App_trace

cited by the Examiner discuss application-level trace logging, which according to paragraph 1 on page 1 is "... a way to enable trace output for an entire application. Enabling Trace at the application level has the effect on enabling Page-level Trace for every page within that application...." While the cited reference seems similar to the claims, a further study of terms in the reference shows that App_trace does not anticipate the claimed material, as will be described below.

App_trace defines an "application" (ASP.NET, Application Overview > What is an ASP.NET Application?, paragraph 1) as "the sum of all files, pages, handlers, modules, and executable code that can be invoked or run in the scope of a given virtual directory (and its subdirectories) on a Web application server." An application has a specific function to perform, and may perform this function for multiple threads or requests concurrently during its lifetime (ASP.NET, Application Overview > Lifetime of an Application, paragraph 1; and ASP.NET, Application Overview > A Note on Multiple Threads, paragraph 1). When Application-trace logging is turned on for an application, "each page in the application will run its page-level trace statements to be output in the client browser" (ASP.NET, page 1, paragraph 2), allowing the trace to record events associated with that application across web pages.

The language in the claims, however, does not refer to an "application." The claims describe tracking a "request," which is defined in the specification on page 2, lines 4-8 as "a request by an entity (e.g,. a client for some task to be performed by another entity (e.g., a server). A request has a finite life time..." A request is a single task that needs to be performed, usually by one or more applications.

The differences between the terms in App_trace and the claims can be illustrated by the analogy of ordering at a fast-food restaurant. At a fast-food restaurant, "applications" would be specific functions, e.g., taking an order, making a burger, packing the food, collecting the money, etc. Thus, an Application-level trace as described in App_trace for "making a burger" would measure how long it took for each step in that function, e.g., putting the frozen patty on the grill, waiting for it to heat, flipping the patty, etc. Note that in the analogy, the application of "making a burger" may handle requests from multiple customers, e.g., three burgers for Customer A, one burger for Customer B, etc. Similarly, Application-

level trace logging in App_trace can handle up to 10 requests at a time (App_trace, page 1, paragraph 3) while logging the performance of each step for each different request that invokes the application.

Continuing the fast-food restaurant analogy, the term "request" as used in the claims would be analogous to an individual customer's order. For example, Customer C orders Request C - a burger, shake and fries. Request C needs to invoke several different applications in order to complete the request: "taking an order," "making a burger," "making a shake," "making fries," "packing the order," etc. The tracking mechanism described by the claims would track and log data for Request C as it moves through these various application stages, with each application working on their functional part of fulfilling Request C.

The claims also refer to tracking "across a request identification boundary in a system" and "storing linking information." These concepts are illustrated by expanding the fast-food restaurant analogy for Request C: "making a burger" is performed independently by Restaurant 1 and within the walls of Restaurant 1, Request C is referred to as "Burger #102." "Making a shake" is performed independently by Restaurant 2 and within the walls of Restaurant 2 is referred to as "Shake #89." "Making fries is performed independently by Restaurant 3 and within the walls of Restaurant 3, is referred to as "Fries #307." The request identification changes across the boundaries of the restaurants in the system. The claims (if applied to the restaurant analogy) would track Burger #102, Shake #89, and Fries #307 as all being part of Request C, and link them together to be able to produce one single receipt with all parts of the order listed on it, timestamps for each function and other such data.

Turning back to the invention, the idea of tracking "across a request identification boundary in a system" is clearly illustrated by Figure 5a. HTTP.SYS, IIS Server, ISAPI, and ASP.NET each receive the request and assign an internal identifier to the request. The request is tracked across these identification boundaries and the linking information for the entire request is stored in the rectangle at the bottom. Note that ASP.NET of Fig. 5a is one of the identification boundaries; ASP.NET runs a specific application to operate on the request and then passes along the request handling to ISAPI. In fact, the ASP.NET Application-trace logging detailed in App_trace could also be invoked in the request scenario of Fig. 5a,

but the log would only contain information pertinent to the *application's* execution in [step 7. Script executes] for this and any other concurrent requests. Application-trace logging, since it is wholly contained in ASP.NET [step 7. Script executes], would not cross identification boundaries as illustrated in Fig. 5a and would not have a method for tracking across those boundaries and linking the information.

Responses to specific claim rejections:

Claim 1: App_trace does not teach "a method for tracking a completion of a request across a request identification boundary in a system including a trace log for recording stages of completing a request." App_trace teaches a method for tracing a log for an application while it services one or more different requests (App_trace Application Overview > A Note on Multiple Threads, paragraph 1) across web pages. In App_trace the request identification does not change across web pages (App_trace, page 2, Session ID). Furthermore, App_trace does not record stages of completing a request since App_trace does not encompass all stages of the request; App_trace application-level trace logging pertains to only the one stage of the request addressed by the specific application. Nor does App_trace store any linking information since App_trace is directed towards the one stage of the request fulfilled by the single application, and the identification does not change within that application .

In order for a claim to be anticipated under §102, the anticipating reference must disclose at least one embodiment that incorporates all of the claimed elements. See, for example, C.R. Bard, Inc. v. M3 Systems, 48 U.S.P.Q.2d 1225, 1230 (Fed. Cir. 1998) ("When the defense of lack of novelty is based on a printed publication that is asserted to describe the same invention, a finding of anticipation requires that the publication describe all of the elements of the claims, arranged as in the patented device..."); In re Bond, 15 U.S.P.Q2d 1566, 1567 (Fed. Cir. 1990) ("For a prior art reference to anticipate in terms of 35 U.S.C. § 102, every element of the claimed invention must be identically shown in a single reference..."). So, in order for claim 1 to be anticipated by App_trace, App_trace must contain every element in claim 1. Since App_trace does not teach tracking a request across identification boundaries, does not include a trace log for recording all stages of a request, and does not store any linking information, App_trace does not anticipate claim 1.

Claims 2-12: Claims 2-12 depend on Claim 1, and therefore incorporate every element in claim 1. Since Claim 1 is not anticipated by App_trace, claims 1-12 are also not anticipated by App_trace.

Claims 13-24, 25-28 and 29-31: These claims respectively teach an event framework, an event utility, and an event record provider which all correspond to the method claims of claims 1-12. Therefore, the Applicants believe claims 13-24, 25-28, and 29-31 are also not anticipated by App_trace based upon the same rationale as for claims 1-12.

CONCLUSION

In view of the above amendments and arguments, the Applicants submit the pending application is in condition for allowance and an early action so indicating is respectfully requested.

If the Examiner has any questions, the Examiner is encouraged to call the undersigned at (312) 474-6300. Applicants believe no fee is due with this response. However, if a fee is due, please charge our Deposit Account No. 13-2855, under Order No. **30835/154731** from which the undersigned is authorized to draw.

Dated: June 12, 2007

Respectfully submitted,

By__/W. J. Kramer/_____
William J. Kramer
Registration No.: 46,229
MARSHALL, GERSTEIN & BORUN LLP
233 S. Wacker Drive, Suite 6300
Sears Tower
Chicago, Illinois 60606-6357
(312) 474-6300
Attorney for Applicants